Research Proficiency Exam:

Putting the curvature back into sparse solvers

Julie Nutini

November 28, 2013

Abstract

Many problems in signal and image processing seek a sparse solution to an underdetermined linear system. A common problem formulation for such applications is the *basis pursuit denoising problem*. Many of the most used approaches for this problem – such as iterative soft thresholding and SPGL1 – are first-order methods. As a result, these methods can sometimes be slow to converge. In this paper, a general two-phase method is presented, which takes advantage of easily-obtainable second-order information for problems that can be expressed as the sum of a convex quadratic function and a closed convex (not necessarily differentiable) function. The details are presented for the use of the general algorithm on a basis pursuit denoising problem, as well as a convex quadratic bound-constrained problem. For the basis pursuit denoising problem, by exploiting the second-order information of the quadratic function, we put the curvature back into sparse solvers and improve upon convergence rates of existing first-order methods. The promise of the proposed method is explored in an application of seismic data interpolation and signal reconstruction.

1 Introduction

We consider the problem

$$\underset{x\in\mathbb{R}^n}{\text{minimize}} \quad f(x)+g(x)\equiv F(x), \tag{1}$$

where $f:\mathbb{R}^n\to\mathbb{R}$ is a convex quadratic function of the form $f(x)=\frac{1}{2}x^\top Hx+b^\top x$, with $H\in\mathcal{S}_+^{n\times n}$ (symmetric positive definite) and $b\in\mathbb{R}^n$, and $g:\mathbb{R}^n\to\mathbb{R}$ is a closed convex (not necessarily differentiable) function with an inexpensive proximal operator.

We propose a two-phase algorithm for problems of the form (1). Because g is not necessarily differentiable, conventional gradient base methods are not applicable. There are generalizations, such as the proximal gradient method, that are applicable to (1). However, these generalizations inherit the slow convergence of the underlying steepest descent methods. Our algorithm combines the *proximal gradient method* with the fast converging *conjugate gradient method* to create an efficient, second-order optimization algorithm.

One formulation of problem (1) replaces the function g with the 1-norm. This formulation is known as the basis pursuit denoising problem (BPDN). Applications of the BPDN problem include compressive sensing [6] and model selection in statistics [7]. Many of the most used approaches to this problem—such as iterative soft thresholding [5] and SPGL1 [4]—are first-order methods. As a result, these methods can be slow to converge. By applying our proposed method to the BPDN problem, we are able to take advantage of the easily-obtainable second-order information of the quadratic term in our use of the conjugate gradient method, and improve upon the convergence rates of existing methods.

The remainder of this paper is organized as follows. In Section 2, we present the details of our two-phase algorithm. In Section 3, we show how our algorithm specializes to the gradient projection conjugate gradient method of Moré and Toraldo [10] when g is the indicator function on the generalized box, forming a large-scale quadratic bound-constrained problem. We then show how our algorithm specializes to the basis pursuit denoising problem when g is equal to the 1-norm. In Section 4, we present the details of our implementation for the case when g is the 1-norm and the results of applying it to a seismic data interpolation example. In Section 5, we summarize our results and present several potential avenues for future work.

2 A Proximal Gradient Conjugate Gradient Algorithm

In this section, we present the proximal gradient conjugate gradient (`pgcg`) algorithm. We start by describing each of the two methods used separately, and then present them together as a complete two-phase method. For the remainder of this paper, we use the notation $\|\cdot\| = \|\cdot\|_2$.

2.1 The Proximal Gradient Method

The proximal gradient (PG) method is a fixed-point iterative method that can be applied to any convex unconstrained optimization problem of the form (1). The PG iteration exploits the inexpensive proximal operator of g. At each iteration, it solves a convex subproblem involving a linearization of f at x. The *proximal mapping* (or proximal operator) for a convex function g is defined as

$$\mathbf{prox}_{\alpha g}(x) = \underset{u}{\operatorname{argmin}} \ g(u) + \frac{1}{2\alpha}\|u - x\|^2, \tag{2}$$

and the *proximal gradient iteration* is given by

$$x_{k+1} = \mathbf{prox}_{\alpha_k g}(x_k - \alpha_k \nabla f(x_k)),$$

where $\alpha_k > 0$ is a step size selected in a line search step to ensure sufficient function value decrease. The following theorem states that the PG method is a fixed-point method.

Theorem 1 (§4.2.1, Parikh and Boyd, [11])**.** *The point x^* solves problem* (1) *if and only if*

$$x^* = \mathbf{prox}_{\alpha g}(x^* - \alpha \nabla f(x^*)) \qquad \text{for any } \alpha > 0.$$

The convergence theory for the PG method proves that given ∇f Lipschitz continuous with constant L and fixed step size $\alpha \in (0, 1/L]$, the PG iteration provides function value decrease at each iteration, and converges to a fixed point of problem (1); see [11] for details. There are variations of this method that allow for L to be unknown.

2.2 The Conjugate Gradient Method

The conjugate gradient (CG) method is a non-stationary iterative optimization method for solving linear systems of the form $Ax = b$, where $A \in \mathcal{S}_+^{m\times n}$. The CG method minimizes the A norm of the error at each iteration, i.e., $\min \|e_k\|_A = \sqrt{e_k^\top A e_k}$, where $e_k = \|x^* - x_k\|$. The following result states that the CG method converges to the exact minimizer of the linear system in a number of iterations less than or equal to the dimension of the problem (assuming exact arithmetic).

Theorem 2 (§7.4, Ascher and Greif, [1])**.** *Given the linear system $Ax = b$ with $A \in \mathcal{S}_+^{m\times n}$, the CG method finds the optimal solution after at most n iterations.*

The CG method is designed for smooth problems. Thus, in order to apply CG to problem (1), we need to consider some smooth restriction of our problem. To do this, we use the idea of an *active set* at a point x with respect to the non-differentiable function g. We define the active set as the set of indices

$$\mathcal{A}(x) = \{i : [\partial g(x)]_i \text{ is } \textit{not} \text{ a singleton}\}. \tag{3}$$

In other words, the active set is the set of indices corresponding to the variables x_i where g is non-differentiable. We denote the set of indices corresponding to the variables that are not active, i.e., those variables that are *free*, by

$$\mathcal{F}(x) = \{i : i \notin \mathcal{A}(x)\} = \{i : [\partial g(x)]_i = [\nabla g(x)]_i\}.$$

By these definitions, g is differentiable with respect to the free variables. Thus, we can use CG to solve a differentiable subproblem in terms of the free variables at each iteration. To formulate a subproblem in terms of the free variables, we consider evaluating the function F at the point $x_k + d$, where d is some search direction in $\mathbb{R}^n$:

$$\begin{aligned}
F(x_k + d) &= \frac{1}{2}(x_k + d)^\top H(x_k + d) + b^\top(x_k + d) && + \quad g(x_k + d) \\
&= \frac{1}{2}x_k^\top H x_k + x_k^\top H d + \frac{1}{2}d^\top H d + b^\top x_k + b^\top d && + \quad g(x_k + d) \\
&= \frac{1}{2}d^\top H d + \underbrace{(Hx_k + b)^\top}_{\nabla f(x_k)} d + \underbrace{\frac{1}{2}x_k^\top H x_k + b^\top x_k}_{\text{constant}} && + \quad g(x_k + d).
\end{aligned}$$

We use a reduced identity matrix, Z_k, to reduce the search direction to the dimensions corresponding to the free variables at the current iteration, i.e., $Z_k = \mathcal{I}[:, \mathcal{F}(x_k)]$, and

$$d = Z_k w = \begin{cases} w_i, & i \in \mathcal{F}(x_k); \\ 0, & i \in \mathcal{A}(x_k). \end{cases}$$

Eliminating the constant term above and making the substitution $d = Z_k w$, we define the following function of w:

$$F_k(w) = \frac{1}{2}(Z_k w)^\top H(Z_k w) + \nabla f(x_k)^\top (Z_k w) + g(x_k + Z_k w).$$

The CG method requires a smooth problem. Hence, the final step in our subproblem formulation is the linearization of $g(x_k + Z_k w)$ about x_k:

$$\begin{aligned} g(x_k + Z_k w) &\geq g(x_k) + \langle g_{x_k}, (x_k + Z_k w) - x_k \rangle \\ &= g(x_k) + g_{x_k}^\top Z_k w, \end{aligned}$$

where $g_{x_k} \in \partial g(x_k)$. Eliminating the constant term $g(x_k)$, we redefine $F_k(w)$ by its quadratic underestimation (modulo constant terms):

$$F_k(w) = \frac{1}{2}(Z_k w)^\top H(Z_k w) + \nabla f(x_k)^\top Z_k w + g_{x_k}^\top Z_k w.$$

Letting $H_k = Z_k^\top H Z_k$ (reduced Hessian) and $r_k = Z_k^\top(\nabla f(x_k) + g_{x_k})$ (reduced gradient of F), we can exploit the available second-order information of the quadratic function f by using a truncated (early terminating) CG method to *approximately* solve the following smooth subproblem in terms of the free variables, i.e.,

$$\underset{w}{\text{minimize}} \quad F_k(w) = \frac{1}{2} w^\top H_k w + r_k^\top w. \tag{4}$$

Remark 1. We note that $Z_k^\top g_{x_k} = [\nabla g(x_k)]_{\mathcal{F}(x_k)}$ is defined and unique because $[\partial g(x)]_i = [\nabla g(x)]_i$ is defined and unique for every $i \in \mathcal{F}(x_k)$. Additionally, if x_k lies in the same face as the solution to problem (1), and w solves (4), then $x^* = x_k + \alpha Z_k w$ solves problem (1) for some $\alpha > 0$.

2.3 Conceptual Algorithm

The pgcg algorithm is a two-phase algorithm. In Phase 1, the PG method is used with a proximal backtracking line search to determine a new working set of the problem, where we define a *working set* by the active variables. This phase sets up the algorithm to form the reduced subproblem for the conjugate gradient method. Once a new working set is determined, we move to Phase 2 and define the reduced subproblem (4). Using a truncated CG method, we find an approximately optimal search direction in terms of the free variables. A backtracking line search is used to update the iteration of the full problem. Since truncated CG is not solved to optimality, there is a possibility that the algorithm would benefit from continuing in Phase 2 with additional CG iterations. To decide, we use

$$\mathcal{B}(x) = \{ i : i \in \mathcal{A}(x) \text{ and } [-\nabla f(x)]_i \in [\partial g(x)]_i \} \tag{5}$$

as the definition of the *binding set* of F at a point x. This is the set of indices corresponding to the active variables that satisfy the optimality conditions of the original problem. In other words, the corresponding variables have reached optimality. If the active set and the binding set at the updated iterate are equal, then we continue to explore the current working set using CG iterations. The intuition is that we may be solving a subproblem in the optimal working set. Thus, we should continue exploiting the fast converging properties of the CG method. If these sets are not equal, then we loop back to Phase 1 and find a new working set.

The line searches used in the pgcg must generalize to a non-differentiable g. We replace the gradient of F at x in a typical line search with the *directional derivative* of F at x along a direction d, defined by

$$F'(x; d) = \lim_{h \to 0} \frac{F(x + hd) - F(x)}{h}.$$

We note that if the function $F(x)$ is differentiable, then the directional derivative is equal to the inner product of d with the gradient of F evaluated at x, i.e.,

$$F'(x; d) = d^\top \nabla F(x).$$

In Phase 1, we use a proximal iteration backtracking line search, which finds an α such that the following sufficient decrease condition is satisfied for some fixed constant $\mu \in (0, 1]$, [letting $p_0 = \mathbf{prox}_g(x_k)$ and $p_k = \mathbf{prox}_{\alpha g}(x_k - \alpha \nabla f(x_k))$]

$$F(p_k) \leq F(p_0) + \mu F'(x_k; p_k - x_k). \tag{6}$$

In Phase 2, the backtracking line search requires the following sufficient decrease condition be satisfied:

$$F(x_k + \alpha d_k) \leq F(x_k) + \mu \alpha F'(x_k; d_k). \tag{7}$$

In both phases, if the sufficient decrease condition is not satisfied, then α is reduced by a factor of 0.5.

We now present a formal description of the two-phase proximal gradient conjugate gradient method for problems of the form (1).

Conceptual Algorithm: [pgcg]

0. **Initialize**: Set $k = 0$ and

    $x_0 \leftarrow \mathbf{prox}_g(x_0)$: starting point
    $\eta_1, \eta_2 \in (0, 1]$: CG and PG sufficient decrease parameters
    $\mu \in (0, 1]$: line search sufficient decrease parameter
    $\tau_{\text{CG}} > 0$: CG optimality tolerance
    $\tau_{\text{opt}} > 0$: optimality tolerance

1. **Phase 1: Determine new working set via proximal gradient iteration.**
    Set $y_0 = x_k$. Generate a sequence $\{y_j\}$ by setting

    $$y_{j+1} = \mathbf{prox}_{\alpha_j g}\big(y_j - \alpha_j \nabla f(y_j)\big)$$

    where $\alpha_j > 0$ is chosen by a proximal iteration backtracking line search so that equation (6) is satisfied. Set x_k equal to the first y_j that satisfies one of the following conditions:

    $$\mathcal{A}(y_j) = \mathcal{A}(y_{j-1}) \tag{8}$$

    OR

    $$F(y_{j-1}) - F(y_j) \leq \eta_2 \max\{F(y_{l-1}) - F(y_l) : 1 \leq l < j\}. \tag{9}$$

2. **Phase 2: Explore current working set via truncated conjugate gradient.**
    Set $w_0 = 0$. Set initial residual $r_0 = Z_k^\top(\nabla f(x_k) + g_{x_k})$, where $g_{x_k} \in \partial g(x_k)$. Generate a sequence of search directions $\{w_j\}$ using a truncated CG method. Set $d_k = Z_k w_j$ for the first w_j that satisfies

    $$\|r_k\|_2 \leq \tau_{\text{CG}} \quad \text{(CG optimality condition)} \tag{10}$$

    OR

    $$F_k(w_{j-1}) - F_k(w_j) \leq \eta_1 \max\{F_k(w_{l-1}) - F_k(w_l) : 1 \leq l < j\}. \tag{11}$$

3. **Update and Loop**:
    Set $x_{k+1} = x_k + \alpha_k d_k$ for some $\alpha_k > 0$ chosen using a backtracking line search such that equation (7) holds. Check optimality condition

    $$\big\|\mathbf{prox}_g\big(x_{k+1} - \nabla f(x_{k+1})\big) - x_{k+1}\big\| \leq \tau_{\text{opt}} \big\|\mathbf{prox}_g\big(x_0 - \nabla f(x_0)\big) - x_0\big\|. \tag{12}$$

    If the above condition holds, STOP.
    Else, if $\mathcal{A}(x_{k+1}) = \mathcal{B}(x_{k+1})$, then set $k = k + 1$ and loop to Phase 2.
    Else, set $k = k + 1$ and loop to Phase 1.

■

The pgcg method combines two well-known algorithms, both of which have supporting convergence theory. Although we do not present the formal convergence theory for the pgcg method, we discuss the structure of the pgcg method and the intuition behind why the pgcg should converge in practice.

In each PG iteration, given α small enough in the line search, the function value of the objective monotonically decreases, i.e.,

$$F\big(\mathbf{prox}_{\alpha g}(x_k - \alpha \nabla f(x_k)\big) < F(x_k).$$

Although the proximal iteration guarantees function value decrease, the progress may be very slow, depending on the conditioning of the problem.

For an n dimensional symmetric positive definite matrix A, we know that the CG method converges (produces the exact solution) in at most n iterations. The truncated CG method provides a monotonically improving sequence of approximations $\{w_j\}$ for the reduced subproblem.

The combination of these two methods does not interfere with the required convergence conditions for either the PG or the CG method; the `pgcg` method simply inherits the convergence of each phase in the active working set. For example, suppose the proximal iteration finds a set of active variables. If the proximal iteration is progressing slowly in terms of sufficient decrease, or has two consecutive iterations with equal active sets, then we propose using the fast converging CG method on a reduced smooth subproblem in terms of the free variables to explore this working set. When truncated CG terminates, it has found an approximately optimal solution in this working set. It is important to note that even if CG only carries out one iteration, then the resulting direction is the direction of steepest descent for the free variables, thus, function value decrease is guaranteed for some $\alpha > 0$. However, it is possible that CG finds a direction that does not result in function value decrease for α greater than some step size tolerance. This is not a problem; as long as the function value is non-increasing in both phases of the algorithm, convergence should hold.

3 Adapting the `pgcg` Method for Specific g

To specialize the `pgcg` method for a specific function g, we simply need to show that there exist corresponding definitions of the following four properties:

- a proximal operator for g, as defined in (2);
- an active set of g, as defined in (3);
- a binding set of F, as defined in (5); and
- an equivalent reduced subproblem of the form (4).

We present the details for two specific functions g: the indicator function on the generalized box and the 1-norm.

3.1 The Gradient Projection Conjugate Gradient Method

The *gradient projection conjugate gradient* was proposed by Moré and Toraldo in 1991, [10]. It solves the *convex quadratic bound-constrained problem*:

$$\underset{x\in\Omega}{\text{minimize}} \quad f(x),$$

where $f(x)$ is a convex quadratic function and $\Omega = \{y : l \leq y \leq u\}$ for some $l, u \in \mathbb{R}^n$. Letting the function g in problem (1) be the indicator function on Ω, we get an equivalent formulation of the above problem:

$$\underset{x\in\mathbb{R}^n}{\text{minimize}} \quad f(x) + \delta_\Omega(x) \equiv F(x). \tag{13}$$

The proximal operator for $g(x) = \delta_\Omega(x)$ is simply the projection operator $P_\Omega(x)$, where by the Projection Theorem

$$p = P_\Omega(x) \quad \Longleftrightarrow \quad p \in \Omega \text{ and } \langle x - p, z - p\rangle \leq 0 \text{ for all } z \in \Omega.$$

(See Theorem 3.14 in [2] for details.) In other words, the proximal operator ensures the current iteration is feasible by projecting it onto the set Ω, resulting in $\delta_\Omega(x) = 0$. To see that $\mathbf{prox}_{\delta_\Omega(\cdot)}(x)$ is equal to $P_\Omega(x)$ for some $x \in \Omega$, consider the proximal mapping

$$\mathbf{prox}_{\delta_\Omega(\cdot)}(x) = \underset{u}{\operatorname{argmin}}\, \delta_\Omega(u) + \frac{1}{2}\|u - x\|^2.$$

Optimality conditions require the condition

$$x - u \in \partial\delta_\Omega(u)$$

be satisfied. The subdifferential of the indicator function on Ω is equal to the *normal cone* to the set Ω, which for any $u \in \Omega$ is defined as

$$N_\Omega(u) = \{n : \langle n, z - u\rangle \leq 0 \text{ for all } z \in \Omega\}.$$

Thus, for any $u \in \Omega$, the optimality condition is satisfied if

$$\begin{array}{rrcll} & \langle x-u, z-u\rangle & \leq & 0, & \forall z \in \Omega \\ \iff & u & = & P_\Omega(x) & \text{(by the Projection Theorem).} \end{array}$$

To define the active set of F, we note that for any point $x \in \Omega$, either $x_i \in (l_i, u_i)$, in which case, $[\partial g(x)]_i = N_\Omega(x_i) = 0$, or

$$[\partial g(x)]_i \in \begin{cases} (-\infty, 0], & \text{if } x_i = l_i, \text{ or} \\ [0, +\infty), & \text{if } x_i = u_i. \end{cases}$$

Hence, according to the definition given in (3), we define the *active set* of (13) at a point $x \in \Omega$ to be the set of indices

$$\mathcal{A}(x) = \{i : x_i = l_i, \text{ or } x_i = u_i\}.$$

Those indices that are not in the active set correspond to the *free variables* of the problem and are defined by

$$\mathcal{F}(x) = \{i : x_i \in (l_i, u_i)\}.$$

According to the definition given in equation (5), the *binding set* of the quadratic bound-constrained problem is the set of active indices that satisfy

$$\begin{array}{rc} & -\nabla f(x) \in N_\Omega(x) \\ \iff & \langle \nabla f(x),\ y - x\rangle \geq 0,\ \forall\ y \in \Omega. \end{array}$$

Analyzing the above inequality component-wise for some arbitrary $y \in \Omega$, we require for each $i \in \mathcal{A}(x)$ that

$$[\nabla f(x)]_i (y_i - x_i) \geq 0.$$

There are 2 cases to consider:

$$\begin{array}{rcccl} x_i = l_i & \Rightarrow & (y_i - x_i) \geq 0 & \Rightarrow & [\nabla f(x)]_i \geq 0 \\ x_i = u_i & \Rightarrow & (y_i - x_i) \leq 0 & \Rightarrow & [\nabla f(x)]_i \leq 0. \end{array}$$

Thus,

$$\mathcal{B}(x) = \left\{ i : \begin{array}{ccc} x_i = l_i & \text{and} & [\nabla f(x)]_i \geq 0; \\ x_i = u_i & \text{and} & [\nabla f(x)]_i \leq 0 \end{array} \right\}.$$

Finally, the reduced subproblem for the convex quadratic bound-constrained problem is equivalent to problem (4), with $r_k = Z_k^\top \nabla f(x_k)$.

3.2 A Sparse Second-Order Method

We now consider the case when $g(x) = \|x\|_1$ in problem (1), forming the basis pursuit denoising problem (BPDN)

$$\underset{x \in \mathbb{R}^n}{\text{minimize}} \quad f(x) + \lambda \|x\|_1 \equiv F(x), \tag{14}$$

where λ is a non-negative regularization constant. The proximal operator for $g(x) = \|x\|_1$ is called *soft thresholding*. As before, we consider the proximal mapping

$$\mathbf{prox}_{\lambda\|\cdot\|_1}(x) = \underset{u}{\operatorname{argmin}}\, \lambda \|u\|_1 + \frac{1}{2}\|u - x\|^2.$$

Observing that the proximal mapping objective is separable, we consider each component individually:

$$\mathbf{prox}_{\lambda|\cdot|}(x) = \underset{u}{\operatorname{argmin}}\, \lambda |u| + \frac{1}{2}|u - x|^2.$$

Optimality conditions require

$$x - u \in \lambda \partial(|u|). \tag{15}$$

Since

$$\partial(|u|) = \operatorname{sgn}(u) = \begin{cases} +1 & \text{if } u > 0; \\ -1 & \text{if } u < 0, \end{cases}$$

when $u \neq 0$, we have

$$\begin{array}{rl} & x - u = \lambda \operatorname{sgn}(u) \\ \iff & x = \lambda \operatorname{sgn}(u) + u \\ \iff & \operatorname{sgn}(x) = \operatorname{sgn}(u). \end{array}$$

This gives us

$$\begin{aligned} & u = x - \lambda \mathrm{sgn}(x) \\ \iff \quad & u = \mathrm{sgn}(x)(|x| - \lambda). \end{aligned}$$

Now, when $u = 0$, $\lambda\partial(|u|) = [-\lambda, \lambda]$. Hence, by equation (15), if $x \in [-\lambda, \lambda]$, then it must be true that $u = 0$. Thus, in the above equation, we take only the positive part of $(|x| - \lambda)$, denoted by a subscript '+'. Thus, we have the component-wise soft thresholding iteration

$$\big[\mathbf{prox}_{\lambda|\cdot|}(x)\big]_i = \mathrm{sgn}(x_i)\big[|x_i| - \lambda\big]_+.$$

We define the *active set* of (14) at x to be the set of indices

$$\mathcal{A}(x) = \{i : x_i = 0\}.$$

It is clear from the above derivation of the soft thresholding iteration that $x_i = 0$ corresponds to the only value of x_i that satisfies the active set definition given in (3). The *free variables* of the problem are then clearly defined by the set of indices

$$\mathcal{F}(x) = \{i : x_i \neq 0\}.$$

To define the *binding set* for problem (14) using equation (5), we require

$$-\nabla f(x) \in \lambda\partial(\|x\|_1).$$

Since the only active indices for problem (14) correspond to when $x_i = 0$, we define the binding set as

$$\mathcal{B}(x) = \big\{i : x_i = 0 \text{ and } [\nabla f(x)]_i \in (-\lambda, \lambda)\big\}.$$

An equivalent formulation of the reduced subproblem (4) for the BPDN problem is attained with $r_k = Z_k^\top \nabla f(x_k) + \mathrm{sgn}(Z_k^\top x_k)$, which is defined, as Z_k eliminates all of the zero entries of x_k.

4 Numerical Results

4.1 Hardware and Software

All testing was performed on a 1.3 GHz Intel Core i5 Macbook Air. We present results for a seismic dataset interpolation example. The implementation was done in MATLAB (v.7.14.0.739, R2012a).

4.2 Initialization, Implementation and Stopping Conditions

Our implementation of the `pgcg` method allows for a cold start, where the initial iterate, x_0, is set equal to zero, or a warm start, where the initialization of x_0 is given as an input. For the following example, we cold start our algorithm.

We employ a different initial step size for each of the line searches used in the `pgcg` method:

$$\begin{aligned} &\textbf{Phase 1}\text{: proximal line search,} \quad && \alpha_0 = \frac{r^\top r}{r^\top H r}, \quad \text{where } r = \nabla f(x_k) \text{, and} \\ &\textbf{Phase 2}\text{: regular line search,} && \alpha_0 = 1. \end{aligned}$$

These choices of α are derived from the minimization of $f(x_k + \alpha d_k)$ over α for different selections of d_k. In Phase 1, $d_k = -\nabla f(x_k)$ and we get the above ratio. In Phase 2, $d_k = Z_k w_k$, and by the conjugacy properties of the conjugate gradient method, α_0 reduces to 1.

We note that matrix-vector products are expensive computations. As is well-known, the CG method only requires one matrix-vector product, $H_k p_j$, for a CG iteration update $w_{j+1} = w_j + \gamma p_j$, where for ease of notation, $\gamma = \gamma_{j+1}$. However, in our truncated CG method, we evaluate the sufficient decrease condition in (11) at each iteration, which requires the calculation of $F_k(w_j + \gamma p_j)$. The following analysis shows that this calculation does not require the evaluation of any additional matrix-vector product:

$$\begin{aligned} F_k(w_j + \gamma p_j) &= \frac{1}{2}(w_j + \gamma p_j)^\top H_k (w_j + \gamma p_j) + (w_j + \gamma p_j)^\top g_k \\ &= \underbrace{\frac{1}{2} w_j^\top H_k w_j + w_j^\top g_k}_{F_k(w_j)} \; + \; \gamma p_j^\top \underbrace{(H_k w_j + g_k)}_{r_{j-1}} + \frac{1}{2}\gamma^2 p_j^\top H_k p_j \\ &= F_k(w_j) \; + \; \gamma p_j^\top r_{j-1} + \frac{1}{2}\gamma^2 p_j^\top \underbrace{H_k p_j}_{\text{previously evaluated}} . \end{aligned}$$

As a failsafe, in addition to the stopping conditions given in Algorithm 2.3, we implement a maximum iteration condition for each step of the pgcg method. For the results presented in the next section, we used the following values:

Step	**Maximum # of iterations**
per PG step	15
per CG step	$\min(n, 20)$
total PG	100
total CG	200

As well, if in either of the back tracking line searches α is reduced so that $\alpha < 10^{-10}$, then the algorithm sets $x_{k+1} = x_k$ and exits the line search. We note that in Phase 1, this results in the equal active sets condition in (8) being satisfied, and the algorithm continues onto Phase 2. In Phase 2, if the line search cannot find a suitable step size for the direction generated by the CG method, then we are either not in the optimal working set, in which case the condition $\mathcal{A}(x_{k+1}) = \mathcal{B}(x_{k+1})$ will not be satisfied, and the algorithm loops to Phase 1, or we are in the optimal working set and the optimality condition (12) is satisfied.

4.3 Results

The following example is an application of seismic data interpolation in sequential source acquisition from a technical report by Kumar, Aravkin, and Herrmann, [9]. For our dataset, we use a frequency slice at 10 Hz of size 354 × 354 from a larger 2D seismic data set from the Gulf of Suez. A restriction matrix operator M is constructed to randomly extract 60% of the columns of the original data set; y represents the vectorized columns of the restricted data set and is referred to as the signal of the data set. We want to find a sparse representation of the signal y in a curvelet operator B. To do so, we must solve the BPDN problem

$$\underset{x}{\text{minimize}}\ \frac{1}{2}\|MB^\top x - y\|_2^2 + \lambda\|x\|_1.$$

The linear system of equations $MB^\top x = y$ is an underdetermined system, and is often ill posed. Hence, a sparse solution x is possible. The solution x is a representation of y in the curvelet operator B. Thus, our final solution is given by $x^* = B^\top x$.

We compare the performance of pgcg against the first-order solver SPGL1, [4]. We note that SPGL1 solves the following basis pursuit problem:

$$\text{minimize}\ \|x\|_1 \quad \text{s.t.}\ \|Ax - b\| \leq \sigma. \tag{16}$$

To ensure that our comparison is valid, we choose a value for the regularization parameter λ that makes the problems (14) and (16) equivalent. To do this, suppose SPGL1 exits with solution x_σ and corresponding residual $r_\sigma = Ax_\sigma - b$. Then the equivalent problem of form (14) requires $\lambda_\sigma = \|A^\top r_\sigma\|_\infty$ (from Lagrange multiplier theory).

In our example, we ran SPGL1 with a basis pursuit solution tolerance and optimality tolerance of 10^{-3}, and a maximum iteration cap of 100. The algorithm terminated with an exit flag of too many iterations. Based on the solution found by SPGL1, we chose $\lambda \approx 16$. Additionally, we used the following pgcg parameter values:

$$\eta_1 = 0.1, \quad \eta_2 = 0.25, \quad \mu = 0.1, \quad \tau_{\text{CG}} = \max\{10^{-4}, 10^{-2}\|r_0\|\}.$$

We recall that η_1 and η_2 are the sufficient decrease parameters used in equations (11) and (9) for the CG method and the PG method, respectively, μ is the sufficient decrease line search parameter used in equations (6) and (7), and τ_{CG} is the CG optimality parameter used in equation (10). Other choices for the parameter values are possible, but these give good results for the given problem.

The pgcg method terminates at an optimal solution for $\tau_{\text{opt}} = 10^{-2}$. The numbers of matrix vector products required by each method are given below:

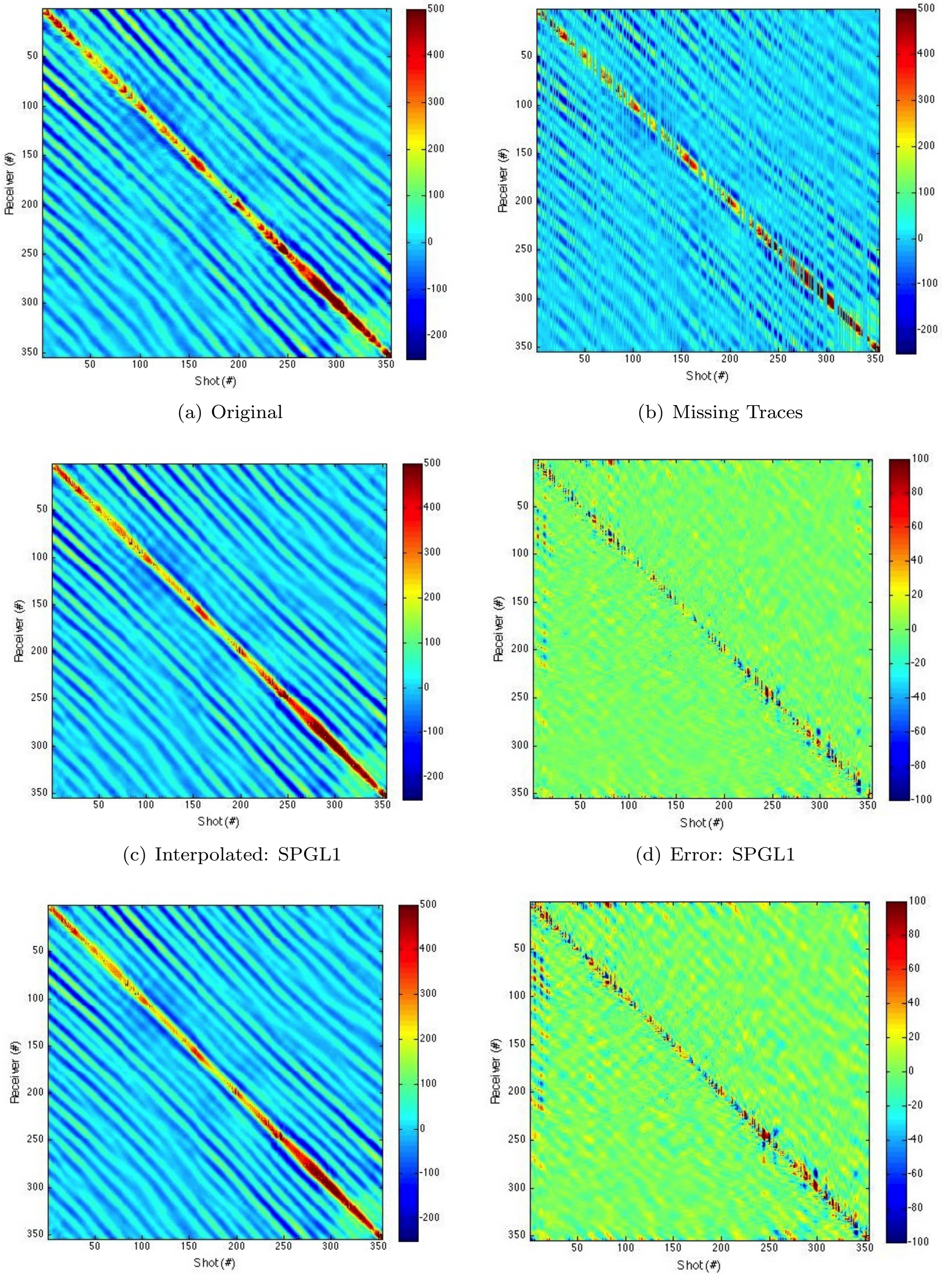
pgcg :	products with $A = 253$,	products with $A^\top = 254$
SPGL1:	products with $A = 139$,	products with $A^\top = 102$.

We can see from the results in Figures 1(c) and 1(e) that both methods interpolated the data well. In both cases, as shown in Figures 1(d) and 1(f), the missing data traces shown in Figure 1(b) were recovered with relatively low reconstruction error except for along the diagonal, where SPGL1 did better than the pgcg method at capturing the details. We note that the results obtained by SPGL1 exactly solve the problem we applied pgcg to. Thus, using an optimality tolerance of 10^{-2}, we do not expect the results of the pgcg method to better the results of SPGL1. Rather, we acknowledge the promise the pgcg method shows with respect to the application of image reconstruction by the results that were obtained.

5 Conclusions and Future Work

We presented a two phase optimization method pgcg for problems that can be expressed as the sum of a convex quadratic function and a closed convex (not necessarily differentiable) function. The first phase employs the proximal gradient method to find a working set of the problem, and the second phase uses the conjugate gradient method to explore the working set by approximately solving a reduced subproblem in terms of the free variables. Specializations of the general method for the convex quadratic bound-constrained problem and the basis pursuit denoising problem are presented. An application in seismic data interpolation is explored, showing the promise of the pgcg method in seismic imaging applications.

There are many avenues of future work for the method presented. In terms of analysis, a formal proof of the convergence of the pgcg method is necessary. The 'hybrid' structure of the pgcg method allows for the extension of using alternative methods in each phase. For example, we could replace the proximal gradient method in Phase 1 with the fast iterative shrinkage-thresholding algorithm [3]. In Phase 2, we could consider alternative exit conditions for the CG method to ensure it is making sufficient progress with respect to the full problem, rather than just the subproblem. For example, rather than completing a backtracking line search after exiting the CG method, we could potentially reduce matrix vector computations by having a check inside the CG method on the progress of the full problem. We touched on the selection of the regularization parameter λ in Section 4.3. Incorporating an active update of λ in the pgcg method, where $\lambda_k \to \lambda$, could lead to faster convergence of the method. Finally, specializing the pgcg method for additional choices of g, as well as adapting the pgcg method to handle complex data are extensions that should to be considered in all of the above suggestions and adaptations.



(a) Original

(b) Missing Traces

(c) Interpolated: SPGL1

(d) Error: SPGL1

(e) Interpolated: `pgcg`

(f) Error: `pgcg`

Figure 1: Gulf of Suez dataset: frequency 10 Hz

References

[1] U. Ascher and C. Greif. A First Course in Numerical Methods. SIAM, 2011.

[2] H. H. Bauschke and P. L. Combettes. *Convex Analysis and Monotone Operator Theory in Hilbert Spaces*. CMS Books in Mathematics. Springer, New York, NY, 2011.

[3] A. Beck and M. Teboulle. A fast iteration shrinkage-thresholding algorithm for linear inverse problems. SIAM J. Img. Sci., 2(2009), 183-202.

[4] E. van den Berg and M. P. Friedlander. *SPGL1*: A solver for large-scale sparse reconstruction. http://www.cs.ubc.ca/labs/scl/spgl1, 2007.

[5] I. Daubechies, M. Defrise, and C. De Mol. An iterative thresholding algorithm for linear inverse problems with a sparsity constraint. Comm. Pure Appl. Math., 57(2004), 1413-1457.

[6] D. L. Donoho. Compressed Sensing. IEEE Trans. Inf. Theory, 52(2006), pp.1289-1306.

[7] B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani. Least angle regression. Ann. Statist., 32(2004), pp. 407-499.

[8] G. Hennenfent and F. J. Herrmann. Application of stable signal recovery to seismic interpolation. In *SEG International Exposition and 76th Annual Meeting*, 2006.

[9] R. Kumar, A. Y. Aravkin and F. J. Herrmann. Fast methods for rank minimization with applications in seismic-data interpolation. Technical Report, 2012.

[10] J. J. Moré and G. Toraldo. On the solution of large quadratic programming problems with bound constraints. *SIAM J. Opt.* (1):93-113, 1991.

[11] N. Parikh and S. Boyd. Proximal algorithms. To appear in *Foundations and Trends in Optimization*, 2013.